Amendments to the Claims

1. (currently amended) A scheduling system for scheduling a plurality of time-dependent tasks, said scheduling system comprising:

(a) an enumerative brute force module;

(b) a dynamic programming module that generates lowest-cost partial paths for said plurality of time-dependent tasks, said dynamic programming module including a hashing function having a low probability of collision, wherein said hashing function is useable in conjunction with a height balanced binary search tree that stores said lowest-cost partial paths;

(c) a genetic module; and

(d) a partitioner module for selecting one of said brute force module, said dynamic programming module, or said genetic module to generate a schedule for selecting said plurality of time-dependent tasks.

2. (original) The scheduling system of Claim 1 further comprising a constraints file, said constraints file providing an input to said partitioner module.

3. (original) The scheduling system of Claim 2, wherein said constraints file includes at least one predetermined time constraint.

4. (original) The scheduling system of Claim 3, wherein said predetermined time constraint further includes at least one constraint selected from the group consisting of:

(a) a precedence constraint, wherein said precedence constraint limits scheduling of said tasks according to a predetermined order of occurrence in time;

(b) a time window constraint, wherein said time window constraint limits scheduling of said tasks within a time window of a predetermined length;

(c) a priority constraint, wherein said priority constraint limits scheduling of said tasks to a sequence according to a predetermined priority; and

(d) a conditional constraint, wherein said conditional constraint limits scheduling of said tasks based on an occurrence of at least one event.

5. (original) The scheduling system of Claim 1, wherein said brute force module includes a task list, a schedule permutation generator, and a schedule evaluator.

6. (original) The scheduling system of Claim 5, wherein said schedule permutation generator includes a first algorithm, said first algorithm enumerating each possible permutation of the time dependent tasks in a schedule.

7. (original) The scheduling system of Claim 5, wherein said schedule evaluator is an efficiency evaluator.

8. (original) The scheduling system of Claim 7, wherein said efficiency evaluator includes at least one parameter selected from the group consisting of a time parameter, a cost parameter, and a user-defined parameter.

9. (original) The scheduling system of Claim 7, wherein said efficiency evaluator includes a second algorithm, said second algorithm determining an efficiency of each task based on a position of said task in a schedule.

10. (original) The scheduling system of Claim 1, wherein said dynamic programming module includes a task list, a dynamic programming schedule permutation generator, and a schedule evaluator.

11. (previously presented) The scheduling system of Claim 10, wherein said dynamic programming schedule permutation generator includes a third algorithm, said third algorithm:

(a) designating a number of tasks to store in memory at a given time;

(b) placing the number of tasks in an order of efficiency; and

(c) determining whether a task outside of the number of tasks is more efficient than a task within the number of tasks.

12. (original) The scheduling system of Claim 10, wherein said schedule evaluator is an efficiency evaluator.

13. (original) The scheduling system of Claim 12, wherein said efficiency evaluator includes a parameter selected from the group consisting of a time parameter, a cost parameter, and at least one user-defined parameter.

14. (original) The scheduling system of Claim 12, wherein said efficiency evaluator includes a fourth algorithm, said fourth enumerating each possible permutation of the time dependent tasks.

15. (original) The scheduling system of Claim 1, wherein said genetic module includes a task list, a genetic schedule permutation generator, and a schedule evaluator.

16. (original) The scheduling system of Claim 15, wherein said genetic schedule permutation generator includes a fifth algorithm, said fifth algorithm mating a first schedule and a second schedule to breed a third schedule.

17. (original) The scheduling system of Claim 15, wherein said schedule evaluator is an efficiency evaluator.

18. (original) The scheduling system of Claim 17, wherein said efficiency evaluator includes at least one parameter selected from the group consisting of a time parameter, a cost parameter, and a user-defined parameter.

19. (original) The scheduling system of Claim 17, wherein said efficiency evaluator includes a sixth algorithm, said sixth algorithm enumerating each possible permutation of the time dependent tasks.

20. (original) The scheduling system of Claim 1, wherein said partitioner module includes a brute force partitioner and a residual evaluator.

21. (original) The scheduling system of Claim 20, wherein said brute force partitioner includes a brute force time completion estimator.

22. (original) The scheduling system of Claim 21, wherein said brute force time completion estimator includes a seventh algorithm, said seventh algorithm:

(a) generating a given number of schedule permutations using the brute force module;

(b) estimating a time to complete all possible schedule permutations using the brute force module based on the time taken to generate the given number of schedule permutations; and

(c) determining whether the estimated time to complete all possible schedule permutations using the brute force module exceeds a given time limit.

23. (original) The scheduling system of Claim 20, wherein said residual evaluator includes a time completion estimator for said dynamic program module and said genetic module.

24. (original) The scheduling system of Claim 23 further including a hardware resource evaluator.

25. (original) The scheduling system of Claim 24, wherein said hardware resource evaluator includes a memory module and a central processing unit.

26. (previously presented) The scheduling system of Claim 23, wherein said residual evaluator includes an eighth algorithm, said eighth algorithm:

(a) generating a given number of schedule permutations using the dynamic programming module;

(b) estimating a time to complete all possible schedule permutations using the dynamic programming module based on the time taken to generate the given number of schedule permutations; and

(c) determining whether the estimated time to complete all possible schedules using the dynamic programming module permutations exceeds a given time limit.

27. (currently amended) A dynamic programming module that generates lowest-cost partial paths for a plurality of time-dependent tasks, said module adapted to provide at least one solution for a scheduling problem, said dynamic programming module including:

(a) a hashing function with a low probability of collision, said hashing function being capable of detecting duplicate solutions generated by said dynamic program module; and

(b) a height-balanced binary tree for providing search, insertion and deletion operations on said lowest-cost partial paths.

28. (currently amended) A scheduling system for scheduling a plurality of time-dependent tasks, said scheduling system comprising:

(a) an enumerative brute force module;

(b) a dynamic programming module that generates lowest-cost partial paths for said plurality of time-dependent tasks, said dynamic programming module including a hashing function with a low probability of collision, said hashing function being capable of detecting duplicate solutions generated by said dynamic program module and a height-balanced binary tree for providing search insertion and deletion operations on said lowest-cost partial paths;

(c) a genetic module;

(d) a partitioner module for selecting one of said brute force module, said dynamic programming module, or genetic module to generate a schedule for selecting said plurality of time-dependent tasks; and

(e) a constraints file, said constraints file providing an input to said partitioner module.

29. (original) The scheduling system of Claim 28, wherein said constraints file includes at least one predetermined time constraint.

30. (original) The scheduling system of Claim 29, wherein said predetermined time constraint further includes at least one constraint selected from the group consisting of:

(a) a precedence constraint, wherein said precedence constraint limits scheduling of said tasks according to a predetermined order of occurrence in time;

(b) a time window constraint, wherein said time window constraint limits scheduling of said tasks within a time window of a predetermined length;

(c) a priority constraint, wherein said priority constraint limits scheduling of said tasks to a sequence according to a predetermined priority; and

(d) a conditional constraint, wherein said conditional constraint limits scheduling of said tasks based on an occurrence of at least one event.

31. (original) The scheduling system of Claim 28, wherein said brute force module includes a task list, a schedule permutation generator, and a schedule evaluator.

32. (original) The scheduling system of Claim 31, wherein said schedule permutation generator includes a first algorithm, said first algorithm enumerating each possible permutation of the time dependent tasks in a schedule.

33. (original) The scheduling system of Claim 31, wherein said schedule evaluator is an efficiency evaluator.

34. (original) The scheduling system of Claim 33, wherein said efficiency evaluator includes at least one parameter selected from the group consisting of a time parameter, a cost parameter, and a user-defined parameter.

35. (original) The scheduling system of Claim 33, wherein said efficiency evaluator includes a second algorithm, said second algorithm determining an efficiency of each task based on a position of said task in a schedule.

36. (original) The scheduling system of Claim 28, wherein said dynamic programming module includes a task list, a dynamic programming schedule permutation generator, and a schedule evaluator.

37. (previously presented) The scheduling system of Claim 36, wherein said dynamic programming schedule permutation generator includes a third algorithm, said third algorithm:

(a) designating a number of tasks to store in memory at a given time;

(b) placing the number of tasks in an order of efficiency; and

(c) determining whether a task outside of the number of tasks is more efficient than a task within the number of tasks.

38. (original) The scheduling system of Claim 36, wherein said schedule evaluator is an efficiency evaluator.

39. (original) The scheduling system of Claim 38, wherein said efficiency evaluator includes a parameter selected from the group consisting of a time parameter, a cost parameter, and at least one user-defined parameter.

40. (original) The scheduling system of Claim 38, wherein said efficiency evaluator includes a fourth algorithm, said fourth enumerating each possible permutation of the time dependent tasks.

41. (original) The scheduling system of Claim 28, wherein said genetic module includes a task list, a genetic schedule permutation generator, and a schedule evaluator.

42. (original) The scheduling system of Claim 41, wherein said genetic schedule permutation generator includes a fifth algorithm, said fifth algorithm mating a first schedule and a second schedule to breed a third schedule.

43. (original) The scheduling system of Claim 41, wherein said schedule evaluator is an efficiency evaluator.

44. (original) The scheduling system of Claim 43, wherein said efficiency evaluator includes at least one parameter selected from the group consisting of a time parameter, a cost parameter, and a user-defined parameter.

45. (original) The scheduling system of Claim 43, wherein said efficiency evaluator includes a sixth algorithm, said sixth algorithm enumerating each possible permutation of the time dependent tasks.

46. (original) The scheduling system of Claim 28, wherein said partitioner module includes a brute force partitioner and a residual evaluator.

47. (original) The scheduling system of Claim 46, wherein said brute force partitioner includes a brute force time completion estimator.

48. (original) The scheduling system of Claim 47, wherein said brute force time completion estimator includes a seventh algorithm, said seventh algorithm:

(a) generating a given number of schedule permutations using the brute force module;

(b) estimating a time to complete all possible schedule permutations using the brute force module based on the time taken to generate the given number of schedule permutations; and

(c) determining whether the estimated time to complete all possible schedule permutations using the brute force module exceeds a given time limit.

49. (original) The scheduling system of Claim 46, wherein said residual evaluator includes a time completion estimator for said dynamic program module and said genetic module.

50. (original) The scheduling system of Claim 49 further including a hardware resource evaluator.

51. (original) The scheduling system of Claim 50, wherein said hardware resource evaluator includes a memory module and a central processing unit.

52. (previously presented) The scheduling system of Claim 49, wherein said residual evaluator includes an eighth algorithm, said eighth algorithm:

(a) generating a given number of schedule permutations using the dynamic programming module;

(b) estimating a time to complete all possible schedule permutations using the dynamic programming module based on the time taken to generate the given number of schedule permutations; and

(c) determining whether the estimated time to complete all possible schedule permutations using the dynamic programming module permutations exceeds a given time limit.

53. (currently amended) A method for scheduling a plurality of time-dependent tasks, said method comprising the steps of:

(a) providing an input to a partitioner module;

(b) selecting a scheduling module from the group consisting of: an enumerative brute force module; a dynamic programming module <u>that generates lowest-cost partial paths for said plurality of time-dependent tasks, said module</u> including a hashing <u>function</u> with a low probability of collision<u>, wherein said hashing function is useable in conjunction with a height balanced binary search tree that stores said lowest-cost partial paths</u>; a genetic module; and

(c) generating a schedule.

54. (currently amended) A method for scheduling a plurality of time-dependent tasks, said method comprising the steps of:

(a) providing an input to a partitioner module;

(b) selecting a scheduling module from the group consisting of: an enumerative brute force module; a dynamic programming module that generates lowest-cost partial paths for said plurality of time-dependent tasks, said dynamic programming module including a hashing function with a low probability of collision, said hashing function being capable of detecting duplicate solutions generated by said dynamic program module and a height-balanced binary tree for providing search insertion and deletion operations on said lowest-cost partial paths; a genetic module; and

(c) generating a schedule.